image and Video Compression with VLSI Neural Networks

Wai-Chi Fang, Senior Member, IEEE

Jet Propulsion Laboratory, California Institute of 'I'ethnology,
MS 3(1(1-241, 4800 Oak Grove 1 Dr., Pasadena CA 91 109-8(199 USA
Tel (818) 354-4695, Fax (818) 393-6943, E-mail: wfang@blacks.jpl.nasa.gov

Bing J. Sheu, Senior Member, IEEE

Department of Electrical Engineering
Signal and Image Processing Institute
University of Southern California, Los Angeles, CA 90089-0271 USA

Abstract An advanced motion-compensated predictive video compression system based on artificial neural networks has been developed to effectively eliminate the temporal and spatial redundancy of video image sequences and thus reduce the bandwidth and storage required for the transmission and recording of the video signal. The VLSI neuroprocessor for high-speed high-ratio image compression based upon a self-organization neural network algorithm is presented in details. Performances of this self-organization network and the conventional algorithm for vector quantization are compared. The proposed method is quite efficient and can achieve near-optimal results. The neural processor includes a pipeline codebook generator and a paralleled vector quantizer which obtains a time complexity O(1) for each quantization vector. A mixed-signal design technique with analog circuitry to perform neural computation anct digital circuitry to process multiple-bit pixel information is used. A 25-dimensional vector- quantizer prototype chip was designed, fabricated, and tested. '1 his neural chip occupies a compact silicon area of 4.6 x 6.8 mm^2 in a 2·0-μm scalable CMOS technology. It provides a computing capability as high as 2 billion connections per second and can achieve an intrinsic speedup factor of 110 compared with a SUN-4/75 workstation.

1 introduction

The goal of image and video compression is to eliminate the temporal and spatial redundancy of video image sequences and thus reduce the bandwidth and storage required for the transmission and recording of the video signal. Broad areas of applications include high-definition television, teleconferencing, remote sensing, radar, sonar, computer communication, facsimile transmission, image database management, and advanced communication and storage systems [1].

Fig. 1 shows an advanced motion-compensated predictive video compression system based on artificial neural networks. The motion-estimation neuroprocessor design is based on a locally connected n~ulti-layer competitive neural network developed for high performance optical flow computing systems [2, 3]. The neuroprocessor design can achieve a high-speed wide range motion estimation. And thus an efficient video motion predictor is made since the motion of regions in the scene is determined so that image points between which the modulation differences are derived can be more accurately chosen. The image compression neuroprocessor design is based on a frequency-sensitive single- layer competitive neural network developed for adaptive vector quantization system [4, 5]. The neuroprocessor design can achieve a high-speed high-ratio image compression by taking advantage of the massively parallel neural computing architecture and VLSI technology. An efficient image compression is therefore achieved since the codevectors in the codebook are adaptively trained from the scene so that the image vectors can be more accurately represented by the index of the most matched codevectors.

In this paper, the frequency-sensitive competitive neural algorithm and its associated VLSI neural processor are presented in details. In section 2, we first describe the frequency-sensitive self-organization algorithm and present the system-level analysis results. In section 3, we then describe a massively paralleled VLSI neural network hardware to implement this algorithm. In section 4, the detailed circuit design and simulation of the neural network processor chip are presented. In section 5, the experimental results of the key building blocks are shown. In section 6, the system-level design of the neural-based VQ system is described.

2 Frequency-Sensitive Neural Learning Algorithm

Shannon's source coding theorem promises that a high compression ratio can be obtained by coding vectors instead of scalars [6]. Vector quantization (VQ) has become a powerful method for speech and image data compression at medium to low bit rates [7]. However, a high-speed VQ adapting to the changing source data statistics is difficult to implement using the popular Linde-Buzo-Gray (LBG) algorithm [8], because it requires the entire training data be processed in a batch mode. Neural network approaches appear to be very promising for intelligent information processing [9-16] due to their massively parallel computing structures and self-organization learning schemes. A number of studies have been reported on using artificial neural networks for VQ applications [10-14]. The basic theory of self-organization networks was presented by Grossberg [11], Kohonen [12], and many other researchers [12-15]. One major challenge of using the simple self-organization network is that some of the neural units may be under-utilized. Various modifications have been proposed to solve this problem [14, 15]. Our frequency-sensitive self-organization (FSO) method modifies Grossberg's variable-threshold competitive learning method [11] by applying a winning frequency and its associated upper-threshold value to the centroid learning rule. It systematically distributes the codevectors in the vector space R^n to approximate the unknown probability density function p(X) of the random training vectors. Codevectors quantize the vector space and converge to vector cluster centroids. This FSO method can produce near-optimal results which will be shown later.

In the 1-loop FSO scheme, the training data is required to pass once in constructing the codebooks. It is a very powerful scheme for adaptive vector quantization due to its relatively low computing requirement and massively parallel computing structure. The 1-loop FSO scheme for adaptive vector quantization is described as follows:

1) Initialize the codevectors and their winning counts $F_i(0)$:

$$\mathbf{W}_i(0) = \mathbf{X}(i) \text{ or } \mathbf{R}(i), \qquad F_i(0) = 1,\ i = 1, \ldots, N \tag{1}$$

where R(.) is a random vector number generation function, $W_i(0) = [W_{i1}(0), W_{i2}(0), \ldots, W_{iM}(0)]$, M is the dimension of codevectors, and N is the number of codevectors.

2) Compute the distortion $D_i(t)$ between an input vector X(t) and all codevectors simultaneously:

$$D_i(t) = d(\mathbf{X}(t), \mathbf{W}_i(t)) = \sum_{j=1}^{M} (X_j(t) - W_{ij}(t))^2, \tag{2}$$

where t is the training time index.

3) Select the distortion-computing neuron with the smallest distortion and set its output $O_i(t)$ to high:

$$O_i(t) = \begin{cases} 1 & \text{if } D_i(t) < D_j(t),\ 1 \le i, j \le N,\ i \ne j, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

4) Update the codevectors with a frequency-sensitive training rule and the associated winning counts:

$$\mathbf{W}_i(t+1) = \mathbf{W}_i(t) + S(t)\,O_i(t)[\mathbf{X}(t) - \mathbf{W}_i(t)], \tag{4}$$

$$S(t) = \begin{cases} 1 & \text{if } 1 \le F_i(t) \le F_{th}, \\ \frac{1}{F_i(t)} & \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

$$F_i(t+1) = F_i(t) + O_i(t), \tag{6}$$

where $S(t)$ is the frequency-sensitive learning rate. Notice that only the winning codevector is updated. The training rule moves the winning codevector toward the training vector by a fractional amount which decreases as the winning count increases. If F_i is larger than an upper-threshold frequency F_{th}, then set $S(t)$ to zero and no further training will be performed at this neural unit.

5) Repeat steps (2) through (4) for all training vectors.

Use of the upper-threshold frequency can avoid codevector under-utilization during the training process for a poorly chosen initial codebook. The selection of the upper-threshold frequency is heuristic and depends on source data statistics and training sequence order. Empirically, an adequate F_{th} is chosen to be 2 to 3 times larger than the average training frequency. In fact, the mean value of index frequencies for codevectors from the LBG method can be used as an upper-threshold frequency.

The performance of the 1-loop FSO method can be incrementally improved by using iteration to adjust codevectors into better cluster centroids. The codebook obtained from the previous iteration is used as the initial values for the current iteration. After the first iteration, the upper-threshold frequency is not needed because a good initial codebook is available. This method is called the multiple-loop FSO method. In the LBG method, the initial codebook could be obtained from the splitting-2 algorithm [8]. The iteration of grouping and calculating centroids in the LBG method is similar to that of updating the closest codevector for each incoming data through the centroid technique in the FSO method. Therefore, the iterative FSO method without the use of upper-threshold frequency asymptotically equals to the LBG method. If the learning process in the FSO method is repeated with the same termination criterion for the LBG method, the result of the multiple-loop FSO method could be similar to that of the LBG method.

The original and reconstructed synthetic-aperture-radar (SAR) ice images using the 1-loop and 2-loop FSO methods for the 10-bit codebook are shown in Fig. 2.1 histograms of the reconstructed images are almost identical to that of the original image. The mean-squared error (MSE) measure is used to evaluate the reconstructed image quality,

$$MSE = \sum_{x=0}^{N_1-1} \sum_{y=0}^{N_2-1} \frac{\| \mathbf{I}(x,y) - \mathbf{I}'(x,y) \|^2}{N_1 N_2} \tag{7}$$

where I is the original image of size N_1 *N2 and I' is its reconstructed image. The MSE values of images using the 1-loop, 2-loop FSO methods, and the LBG method are listed in "Table 1 and also plotted in Fig. 3. The performance figures given for the three algorithms are obtained under the same condition by training the same picture and being measured with the MSE criterion. Performance of the FSO methods is very close to the LBG method. The reconstructed images using the FSO method on 5x5 subimage blocks are reasonably good.

The large dynamic range of images requires that the effective compression algorithms should be adaptive to the local image statistics. For the vector quantization approach, edge degradation is very severe if no adaptation is allowed for different scene characteristics. If the codebook trained from the FSO method for the SAR ice image shown in Fig. '2(a) is used to encode and decode the Girl image without any modification, the mean-squared error is 1063. After training this poor codebook by using the FSO method, a much smaller MSE value of 47 can be achieved. This result illustrates that the codebook can be successfully adjusted according to the statistic change of source data.

In a high-speed hardware design, the resolution limit for synapse arrays by analog circuitry is a very important factor. The simulation results from the FSO method for different signal resolutions show that performance of the 8-bit resolution case is reasonably close to that of floating point computation. If the codebook size is large, the performance for the 6-bit resolution and that for the floating point computation are not distinguishable. In a larger codebook, each codevector is trained from a smaller portion of source data and thus the error induced by finite resolution is also smaller.

3 VLSI Neural Processor Architecture

The proposed VLSI neurocomputing architecture for adaptive image comp ression using frequency-sensitive self-organization network is shown in Fig. 4. The FSO network consists of two layer: an input layer and a competitive layer.

The input layer consists M input neurons which correspond to the elements of the M-dimensional input vector. Each input neuron gets its input from the external world and distributes the buffered signal to N distort ion-computing neural units in the competitive layer. Each distor tion-computing neuron calculates a square of Euclidean distance helm'cwn its codevector and the input vector. The competitive process is performed throughout the whole layer by the winner-take all operation. The winning neural unit is determined according to the minimum distor tion criterion. The synapse weights are then updated according to FSO learning rule as specified in the (4), (5), and (6). The learning rule can be implemented in software by a host processor or in a dedicated digital signal processing (1)SI') chip by an algorithm-specific hardware design.

By using the massively paralleled neural computing paradigm and the mixed-signal VLSI design technique, the FSO network can be implemented on a chip set. The block diagram of a VI .SI design of the FSO neural processor is shown in Fig. 5. The high-level functional blocks of this neural processor include an analog vector quantizer chip and a digital codebook generator chip,

For the analog vector quantizer, the mixed-signal VLSI design technique with the analog circuitry to perform massively parallel neural computation and digital circuitry to process multiple-bit pixel information is-used. The analog vector quantizer realizes a full-search vector quantization process for each input vector at a time complexity 0(1). It consists of the input neurons, programmable synapse matrix, summing neurons, winner-take-all cells, and an index encoder. The programmable synapse matrix is composed of $M \times N$ synapse cells which correspond to N M-dimensional codevectors. The output neuron array is composed of N summing neurons which perform paralleled summation of the distortions between the input vectors and codevectors. The winnct-take all block consists of N competitive cells which perform paralleled comparison among N inverted distortion values and choose a single winner. It also provides the sufficiently high output level for the winning node against the rest. The index encoder is a N-*to n* demultiplexer which uses binary codes to encode the N classes.

For the digital codebook generator, the custom I)SI' circuit design can be used to achieve high-per formance requirements. The digital codebook generator is specialized to implement the FSO training rule in time comp lexity $O(1)$ for each input vector. It is a co-p rocessor module to support high-speed neural networ k learning algor ithm. The digital codebook generator consists a I)SI'-based trainer, a dual-port vector memory, and a timing/control block. It uses digital n-bit index of winning neuron gener ated by analog VQ to access the corresponding winning codevector and fr equency. It updates the codevector of the winning neural unit anti then increases the associated winning frequency by one. The updated codevector is wt itten to both the digital codebook memory and the analog synapse array. The codebook memory is built with 2-port dynamic memory anti organized as N-word hy 8* M-bit to reduce I/O communication traffic. An incremental adaptation of the codebook is performed in a read-modify-write cycle only whenever there is a winning codevector chosen. The digital address control block is also shared hy the analog vector quantiz er to address the cot responding synapses for codevector loading, modification, and refreshing. The digital input vector is converted into the analog value using the digital-to-analog (i)-to-A) converter array and fed to input neurons of the analog vector quantizer.

4 Detailed Circuit Implementation

Advanced studies to improve the circuit performance and to reduce the area/power of the neural building blocks are essential to implement the highly complex neural systems in VLSI technologies [16]. Computer simulation and laboratory experiments on these neural circuits have been conducted. Figure 6 shows one slice of the FSO network consisting of key circuit blocks. The simulated processing time for one network iteration is about 250 ns. Each iteration cycle includes input buffering, synapse multiplication, neuron summing, winner-take-all operation, and index encoding. The load capacitances for each block is estimated and included. The major delay of the network is at the input neuron due to the large load capacitance associated with the long signal wire. Transistor sizes of the input neurons can be increased to reduce the delay time.

4.1 Input Neuron

In the input layer, the input neuron design is an unity-gain buffer. The input signal is composed of M input lines and each input line is applied to one row of N synapse cells. The load capacitance for the input neuron is quite significant. It is estimated to be 5 pF for the N=64 case. Thus the input voltage needs to be buffered before it is distributed to the synapse cells. The input neuron is a conventional operational amplifier in a unity-gain configuration. The experimental input neuron has a DC gain of 95.82 dB and an unity-gain frequency of 10 MHz at a load capacitance of 5 pF. The settling time to within 0.1% accuracy is 80 nsec for the 1 Vp-p input pulse.

4.2 Output Summing Neuron

In the VLSI chip, the output summing neuron converts the total current into the output voltage, which is applied to the winner-take all circuit. The output of each neuron is the distortion measure and the minimum among them is to be chosen as the winner in the competitive layer. Since the WTA circuit selects the maximum input, the output neuron has its summed current converted into the voltage with the sign reversed. Current summation occurs at every column of the synapse matrix. To ensure the linear current-to-voltage conversion for a wide operation range, the output neuron can support the summing current up to 1 mA. It has the large output buffer to allow the large amount of current. The settling time to within 0.5% accuracy is 25 nsec for 0.8 mA input current and 2 pF load capacitance. Linear resistance is required to convert the current into the voltage. Since the accuracy of a passive resistor is very low (could be up to 20% error) without the additional trimming technique. In addition, to achieve the resistance value of 2 KΩ and support large current flow value (up to more than 1 mA) , a relatively large area is used even using the well region with high sheet resistance. The MOS transistors biased in the triode region can be used to synthesize the linear resistance [17]. The layouts of the current summing neuron and the linear floating resistor occupy 116 λ x 228 λ and 116 λ x 64 λ, respectively.

4.3 Programmable Synapse

The programmable synapse design is a simple and modified wide-range Gilbert multiplier [16] which can perform real-valued multiplication in four quadrants and achieve an 8-bit precision. In order to realize the function specified in (2), the synapse cell calculates the square of the difference between the input voltage and the synapse value. Figure 7 shows the circuit schematic of the synapse cell and the size of each transistor. The layout of one synapse occupies 112 λ x 82 λ in the MOSIS scalable CMOS design [20].

In order to achieve a wide operation range, the differential pair for (V_1-V_2) and (V_3-V_4) are separated using the current mirror circuitry. The output current is obtained from the cascade current mirror stage consisting of transistors M_{21} through M_{24}. It is approximated by

$$I_{out} = \sqrt{\frac{k\beta_1\beta_{11}}{2}}\,(V_1-V_2)(V_3-V_4), \tag{8}$$

transistor $M_{3(4)}$ to transistor $M_{13(16)}$, and β_1 and β_{11} are the M_1 and M_{11} transistors, respectively. Figure 8 shows the DC

characteristics of the multiplier. The linearity error is less than 2% for an input voltage range of -1.8 V to 1.8 V for both inputs of (VI - V_2) and (V_3-V_4). The center of the parabolic curves are shifted by the amount of the weight values. In order to calculate $(X_i\text{-}W_{ji})^2$, the input voltage are rearranged. When Xi is applied to V_1, V_3 and W_{ji} is applied to V_2, V_4, then (8) becomes

$$I_{ji} = \sqrt{\frac{k\beta_1\beta_{11}}{2}}\,(X_i\text{-}W_{ji})^2 = \sqrt{\alpha_{ji}}\,(X_i\text{-}W_{ji})^2, \textit{ for } 1 \le i \le M, \textit{ and } 1 \le j \le N. \qquad (9)$$

The input voltage X_i is fed into the synapse cell through the input neuron. The synapse value W_{ji} is dynamically stored on the capacitance of the MOS transistors. It must be refreshed periodically since a parasitic leakage exits in the diffusion-to-substrate junction. From the measured charge retention characteristics, a refresh cycles of 0.5 sec is sufficient for the 8-bit synapse accuracy. Since there are $M \times N$ synapse cells, the required speed should be higher than $2MN$ Hz for one D-to-A converter to refresh all the synapse cells. In the prototype chip, there are 1600 synapses and the required refresh period is about 31 msec.

4.4 Winner-1'ake-All Cell

The performance of the WTA circuit built with transistors biased in the subthreshold region [16] has limited performance due to the inherently low speed operation and a small noise immunity. Our high-precision WTA circuit operates in the strong inversion region and can provide fully binary output values which are easily interfaced with digital circuitry for network learning, It can also be extended for a large network of over 1,0?4 inputs in real-world applications. Our analog WTA circuit can determine the winning unit at one cycle instead of $\log_2 N$ clock cycles using MAXNET [10].

The WTA circuit schematic and the size of each transistor arc shown in Fig. 9. The layout area of each WTA cell is 58 λ x 96 λ. One WTA cell consists of two portions. The first portion converts input voltage into the current which is compared and redistributed in the common signal line. In the second portion, the current is converted into the output voltage. All transistors operate in the saturation region. V_{CM} is the common node voltage to which all source terminals of input transistors M_1 are connected. As the number of inputs increases, the circuit can be extended by abutting this common signal node from the cells. Through this node, the total bias current is contributed by every cell. Since the source terminal is at a common voltage for all the cells, the current flowing through each cell is proportional to V_i^2. Thus, the largest input can fetch the largest current out of the total bias current. This largest current can make the corresponding output saturated at the positive supply voltage value. On the other hand, the other outputs will be saturated at the negative power supply value. The total bias current is provided by the transistor M_5 of the cells. Instead of fixing the amount of the total bias current outside the cells, each cell provides its own share of the bias current. Since the bias current increases in proportion to the number of inputs, the circuit response time is independent of the number of inputs. Figure 10 shows t he measurement results of a X3(-input WTA structure.

5 Prototype Neural Chips

The prototype neural chip design for a 25-dimensional adaptive vector quantizer of 64 codevectors has been implemented in a silicon area of 4.6 *mm* x 6.8 *mm* using the 2-pin CMOS technology from the MOSIS service. This neural network based vector quantizer (NNVQ) chip employs a mixed analog-digital configuration The power line from analog and digital blocks are separated to avoid noise coupling from digital parts to the highly sensitive analog parts. The die photo of the NNVQ chip is shown in Fig. 11.

in the analog neural chip design, the common-centroid layout technique greatly alleviates the device mismatch effect [17]. Use of large devices can also reduce sensitivity to mobility variation and channel-length modulation variation [18]. The carefully chosen device sizes of Fig. 1? help to keep the variation of the transconductance constant b below 1 % [19]. In addition, the operational dynamic range of the synapse cell can be increased with larger devices, which can ensure the 8-bit accuracy for the analog circuitry. The dimensionality of the NNVQ design is 25 which is functional based on our

measurement. "This chip is also extendible to implement VQ of a larger codebook. An adaptive vector quantizer of 1,024 codevectors can be implemented by cascading 16 such prototype chips or using a larger clip through a submicron fabrication technology. An adaptive vector quantizer of 1,024 codevectors can be designed in a single chip of 125 mm^2 silicon area using 1-μm CMOS technology.

An image compression system can be constructed! with the analog NNVQ chip and a digital training co-processor. The system throughput is about 2 million vectors per second. It can meet a broad range of high-sped applications such as HDTV which requires a throughput rate of 1.2 million vectors per second for 25-dimensional VQ on 1,024-pixel x 1,024-pixel images.

6 ('occlusion

An advanced motion-compensated predictive video compression system based on artificial neural networks has been developed to effectively eliminate the temporal and spatial redundancy of video image sequences and thus reduce the bandwidth and storage required for the transmission and recording of the video signal. The VLSI image compression neuroprocessor based upon a frequency sensitive self-organization neural algorithm is described in details. The efficiency of this FSO neuroprocessor is measured by its compression ability, the resulting distortion, error tolerance, and the implementation simplicity. The digital co-processor for the codebook training is also described for a custom DSP circuit design. By using a mixed analog-digital design approach in the massively parallel computation blocks, the advantages of small silicon area, low power consumption, and reduced I/O requirement can be achieved. A 25-dimensional vector quantizer of 64 codevectors has been implemented in the NNVQ prototype chip. This chip occupies a compact silicon area of 4.6 x 6.8 mm^2 in a 2.0-μm scalable CMOS technology. Its throughput rate is 2 million vectors per second and its equivalent computation power is 2 billion connections per second.

Acknowledgments

The research described in this paper was partially carried out by the Jet Propulsion laboratory, California institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] A.K. Jain, "Image data compression: a review," Proc. IEEE, vol. 69, no. 3, pp. 349-389, Mar. 1981.

[2] W.-C. Fang, B. J. Sheu, "Real-tinle computing of optical flow using adaptive VLSI neuroprocessors," *IEEE International Conference of Computer Design*, pp. 122-125, Cambridge, MA, Oct. 1990.

[3] J.-C'. Lee, B. J. Sheu, W.-C. Fang, "VLSI neuroprocessor for video motion detection," *IEEE Trans. on Neural Networks*, vol. 4, no. 2, pp. 178-19, Match 1993.

[4] W. Fang, B. Sheu, O. T. Chen, "A neural network based VLSI vector quantizer for real-time image compression," J. A. Storer, J. H. Reif (Ed.), Proc. 1991 IEEE Data Compression Conference, IEEE Computer Society Press: Los Alamitos, CA, 1991.

[5] W.-C Fang, B. J. Sheu, O. T.-C. Chen, J. Choi "A VLSI neural processor for image data compression using self-organization networks," *IEEE Trans. on Neural Networks*, vol. 3, no. 3, pp. 506-518, May 1992.

[6] C. Shannon, "A mathematical theory of communication," Bell Systems Tech Jour, vol. ?7, pp. 379-423, 1948.

[7] A. Gersho, R. Gray, Vector Quantization ans Signal Compression, Kluwer Academic Pub., Boston, MA, 1992.

[8] Y. Linde, A. Buzo, R. M. Gray, "An algorithm for vector quantizer design," IEEE Trans. on Comm., vol. COM-28, pp. 84-95, Jan. 1980.

[9] D.E. Rumelhart, J. L. McClelland, Parallel Distributed Processing, MIT Press: Cambridge, MA, 1986.

[10] R. R. Lippmann, "An introduction to computing with neural nets," IEEE Acostics, Speech, and Signal Processing Maganize, vol. 4, no. 2, pp. 4-22, Mar. 1987.

[11] S. Grossberg, 'T'repetitive learning: From interactive activation to adaptive resonance," Cognitive Sci., vol. 11, pp. 23-63, 1987.

[12] 1'. Kohonen, Self-Organization and Associative Memory, 2nd Ed., Spring-Verlag: New York, NY, 1988.

[13] N. M. Nasrabadi, Y. Feng, "Vector quantization of images based upon the Kohonen self-organizing feature maps," Proc. 1988 International Joint Conference on Neural Networks, vol. 1, pp. 101-1CW, San Diego, CA, June 1988.

[14] S.C. Ahalt, A. K. Krishnamurthy, P. Chen, D. Melton, " Competitive learning algorithm for vector quantization," Neural Networks, vol. 3, pp. 277-290, 1990.
[15] D. DeSieno, " Adding a conscience to competitive learning," Proc. 1988 International Joint Conference on Neural Networks, vol. I, pp. 117-124, San Diego, CA, June 1988.
[16] C. Mead, Analog VLSI and Neural Systems, Addison Wesley: New York, NY, 1989.
[17] R. Gregorian, G. Temes, Analog MOS Integrated Circuits for Signal Processing, Wiley-Interscience:, NY, 1986.
[18] P. R. Gray, R. G. Meyer, Analysis and Design of Analog Integrated Circuits, John Wiley & Son: NY, NY, 1984.
[19] K. R. Lakshmikumar, R. A. Hadaway, M. A. Copeland, " Characterization and modeling of mismatch in MOS transistor for precision analog design," IEEE J. Solid-State Circuits, vol. SC-21, no. 6, pp. 1057-1066, Dec. 1986.
[20] C. Tomovich, "MOSIS - A gateway to silicon," IEEE Circuits and Devices Mag. vol. 4, no. ?, pp. 22-23, Mar. 1988.

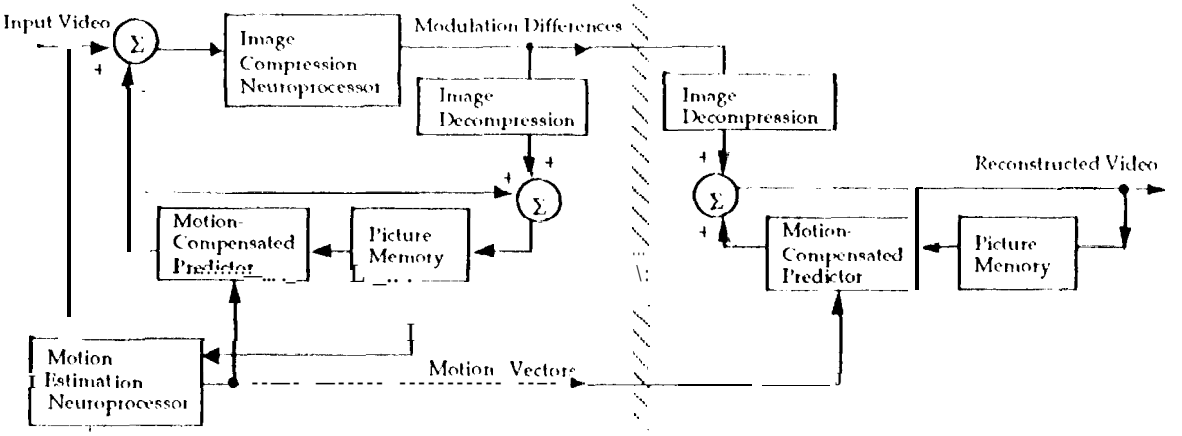


Fig. 1 Advanced motion-compensated predictive video compression system with neural networks

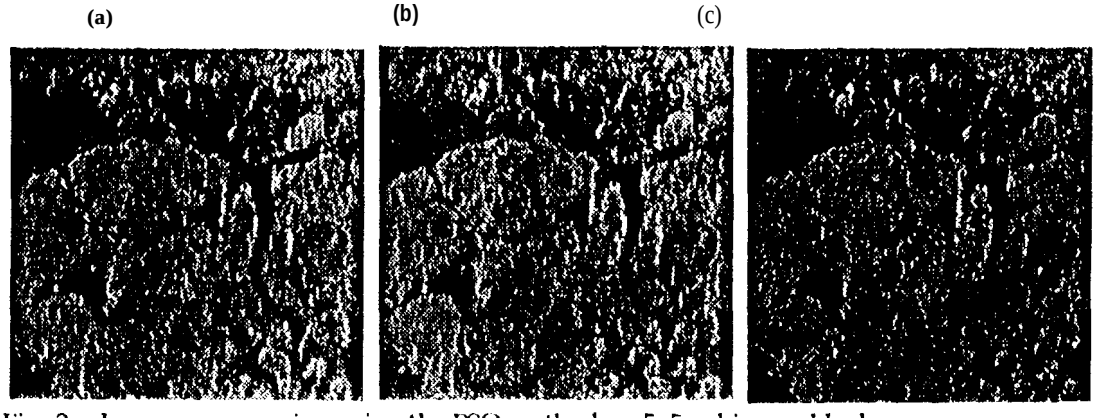


Fig. 2 Image compression using the FSO method on 5x5 subimage blocks.
(a) Original SAR Ice image.
(b) Reconstructed image using 10-bit one iteration FSO codebook; MSE = 86.31.
(c) Reconstructed image using 10-bit tw(~-iteration FSO codebook; MSE = 82.35

Table 1 Performance comparison of VQ methods.

Algorithms / Codebook Size	one iteration FSO			two iteration FSO			LBG	
	MSE	SDR	*Uh*	MSE	SDR	*Uh*	MSE	SDR
10-bit	86.31	18.16	20	82.35	18.37	20	78.64	18.61
9-bit	101.36	17.47	40	97.89	17.62	40	92.84	17.89
8-bit	115.04	16.92	80	112.08	17.03	80	106.95	17.28
7-bit	127.51	16.47	160	124.41	16.58	160	119.80	16.78
6-bit	142.27	16.00	320	138.56	16.11	320	133.78	16.30

Note: MSE: Mean-Squared Error; SDR: Signal-to-Distortion Ratio; *Uh*: Upper Threshold Frequency.

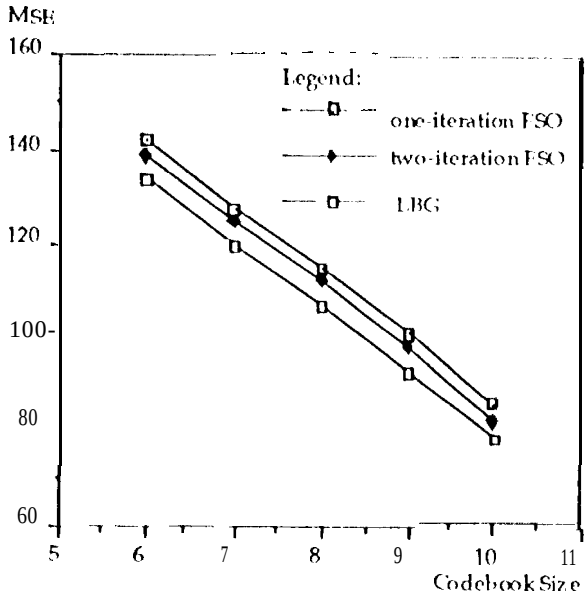


Fig. 3 Mean-squared errors of image compression using the FSO and the LBG on 5x5 subimage blocks of the 512x512-pixel SAR Ice image.

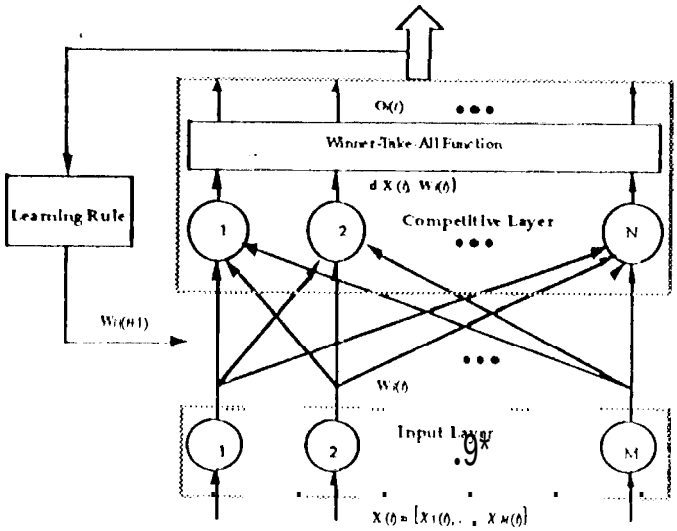


Fig. 4 Structure of the FSO neural network.

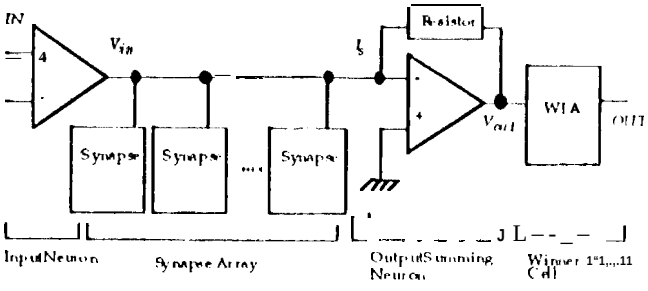


Fig. 6 Block diagram of the FSO network slice,

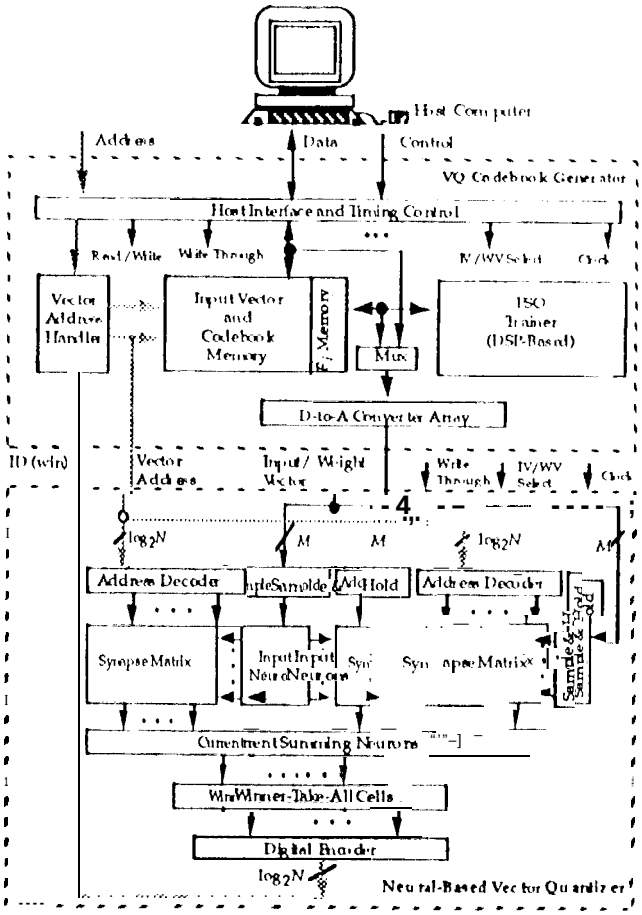
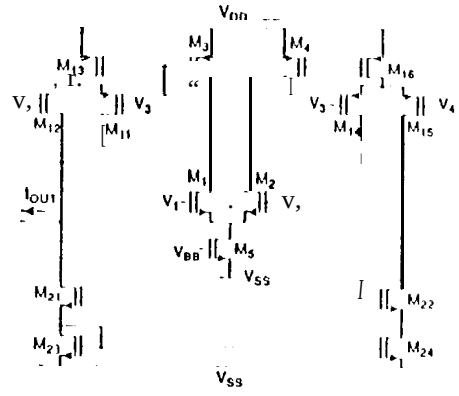


Fig. 5 Block diagram of the FSO neuroprocessor.



Transistor	$M_{1,2}$	$M_{3,4}$	M_5	$M_{11,12,13,15}$	$M_{13,16}$	$M_{21,22}$	$M_{23,24}$
Size (μm/μm)	4/32	10/3	12/4	4/22	21/3	4/8	4/12

Fig. 7 Circuit schematic and transistor sizes for the programmable synapse.

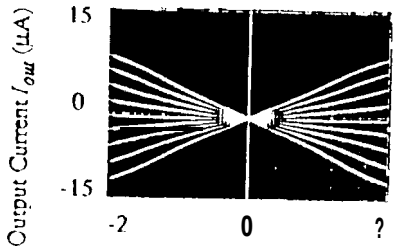


Fig.8 Measured synapse output current versus neuron input voltage $(V_1 - V_2)$ with different weight voltage values $(V_3 - V_4)$: -?,(), -1.0, -()..5, -0.25, 0,0.? 5,0.5, 1.0, and 2.0 V (bottom to top).

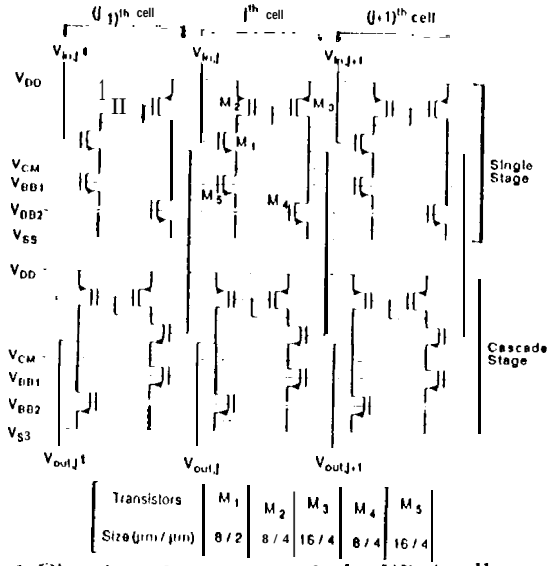


Fig.9 Circuit schematic of the WTA cells.

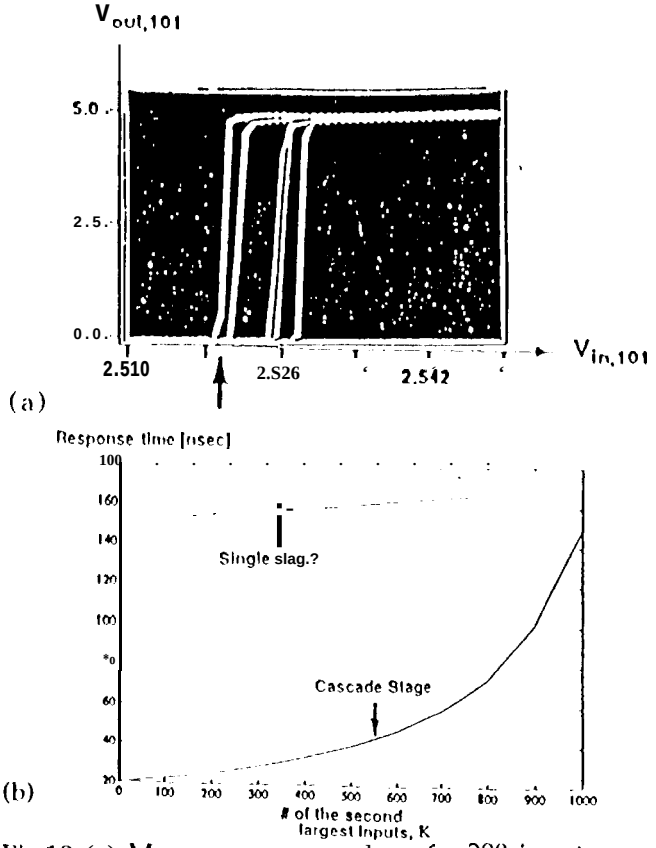


Fig.10 (a) Measurement results of a 200-input WTA test structure. Output voltages of the winner with different numbers of cells having the second largest input: 2, 50, 100, 150, and 199, from left to right.
(b) Response time of the winning output on a 1000-input WTA with different numbers of cells having the second largest input voltage values.

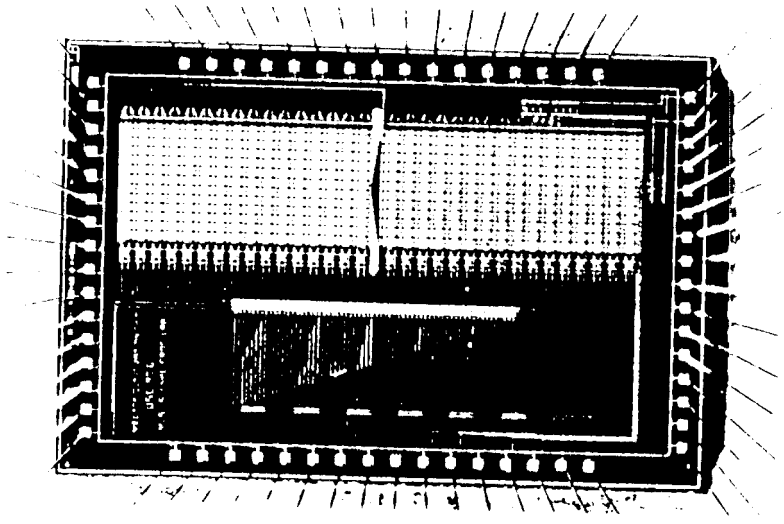


Fig.11 Die photo of the NNVQ prototype chip.